2006年3月号付属
CPU基板関連企画

ARMベース・システムLSI開発の事例研究

デバイスの記事　付属基板関連

――CPUの選択，バス構成，グラフィックス処理やビデオ表示制御の取り扱い

山崎尊永

ここではARM926EJ-SをコアとするグラフィックスLSIの開発事例を紹介し，ARMベースのシステムLSIを開発する際のポイントを解説する．ファブレス半導体メーカである筆者らは，ビデオ表示やグラフィックス描画の機能，および各種入出力インターフェースを備えるLSIを開発した．フレーム・メモリをピンポン・バッファとして使える構成にしたり，グラフィックス描画のアクセラレータなどを組み込んだ．（編集部）

CPUコアを内蔵するシステムLSIやASIC（application specific integrated circuit）を開発するとき，CPUを選定するポイントとして，以下の項目があるかと思います．

- 性能と消費電力
- 手持ちのOSやソフトウェア資産の継承性
- OS，ミドルウェア，ライブラリ，アプリケーション・ソフトウェアの充実度
- ソフトウェア開発環境の充実度（コンパイラやICEなど）
- LSI内部のシステム設計の容易性
- CPUコアを取り巻くIP（intellectual property）コア群の充実度
- 各種技術情報の豊富さ
- LSIとしての最適コストの実現や最適技術の活用のため，複数のASICベンダやファブなどから提供されていること

CPUの選定は，性能や消費電力はもちろんのこと，アーキテクチャへの慣れやソフトウェアの継承性も重要であり，こうした条件はアプリケーションごとに異なるため，「これがベスト」というものはありません．ただし，初めてCPUを選定するときや，これまでの流れとは異なるアプリケーション向けのLSIを開発する場合には，上記の条件を比較的よく満たしているCPUの一つとしてARMを選択することが考えられます．

ファブレス半導体メーカである筆者ら（アプローズテクノロジーズ）は，ARM926EJ-SをコアにしたグラフィックスLSI「AP4010」を開発しました．この開発事例を紹介しながら，ARM9コアの使用上のポイントや内部バスの構成，グラフィックス制御方式などについて説明します．

● 仮想記憶対応OSの利用を考慮してARM926を選択

システム開発を行っている顧客から筆者らの会社に，シンプルなハードウェアでテレビやLCD（液晶ディスプレイ）パネルに2次元グラフィックスを表示させたい，という要求がありました．各方面へのマーケティング活動の結果，一つのLSI上にCPUやグラフィックス制御機能，各種ユーザ・インターフェース，通信機能などを組み込んだシステムLSIを開発することを決めました．

このLSIは，TFT（thin film transistor）液晶パネルやSTN（super twisted nematic）液晶パネルのLVCMOS信号，およびテレビのコンポジット・ビデオ信号を直接出力し，グラフィックス描画のアクセラレータや液晶タッチ・パネルへの直結インターフェース，USBインターフェースなどを内蔵しています．プログラムやデータを格納するためのROMとSDRAM（synchronous DRAM）をこのLSIに外付けするだけで，各種のユーザ・インターフェースを備えるグラフィックス・システムを構築できます．

CPUについては，映像・音声系のミドルウェアを動かす

KeyWord　ARM926EJ-S，AP4010，マルチレイヤAHB，ETM9，V5TEJアーキテクチャ，ピンポン・バッファ，BitBlt，アファイン変換，アルファ・ブレンディング，MMU

表1　開発したグラフィックスLSI（AP4010）の仕様
本稿では，おもに陰の付いているところを説明する．

機　能	項　目	内　容
CPU	CPUコア	ARM926EJ-Sコア
	動作周波数	CPU：108MHz，内部AHBバス：108MHz
	MMU	あり
	TCM	ITCM：8Kバイト，DTCM：16Kバイト
	キャッシュ	命令キャッシュ：8Kバイト，データ・キャッシュ：8Kバイト
	デバッグ・インターフェース	JTAG端子，ETM9インターフェース
割り込み	外部要因	FIQ×1本，IRQ×2本
	内部要因	内蔵周辺ブロックからの割り込み（23本）
	割り込み優先順位	16レベル（4ビット指定）
外部メモリ・インターフェース	データ・バス幅	16ビット
	SRAM/フラッシュ・メモリ空間	5エリア（SRAM，NOR，NAND直結インターフェース 低速I/Oデバイス接続サポート）
	SDRAM空間	1エリア（SDRAM（108MHz）直結インターフェース）
ビデオ表示プロセッサ（VDC）	テレビ出力	NTSC/PALコンポジット・ビデオ信号出力 解像度指定可能（VGA/QVGA）
	TFTパネル出力	LVCMOS TFTパネル・インターフェース信号（RGB＋同期）出力 QVGA，VGA，SVGAクラスの任意サイズ指定可能
	STNパネル出力	LVCMOS STNパネル・インターフェース信号（8ビット）出力 QVGAクラスのパネルに対応
	色深度	32,768色（15ビット）
	サポート機能	ハードウェア・スクロール機能，テレビとLCDの同時表示可能 多面フレーム・メモリ対応（ピンポン・バッファ）
グラフィックス描画プロセッサ（GDP）	BitBlt転送	フレーム・メモリ空間への高速DMA描画
	アファイン変換	移動・拡大・縮小・回転のハードウェア・サポート
	アルファ・ブレンディング	半透明描画処理
	色変換	原画の色を変換して描画
そのほかの周辺機能	クロック	27MHz水晶発振器，PLLてい倍回路
	電力制御	スタンバイ，ストップ，ディープ・ストップ
	HDMAC	2チャネル（うち1チャネルは外部転送要求受け付け可能）
	オーディオA-Dコンバータ（AAD）	10ビットA-Dコンバータ，オーバサンプリング機能，DMA機能
	オーディオD-Aコンバータ（ADA）	12ビットD-Aコンバータ，ステレオ出力可能
	ソフトウェア用タイマ（TMRS）	リアルタイムOS用16/32ビット・タイマ×2本
	ハードウェア用タイマ（TMRH）	16ビット・タイマ×2本（パルス出力，パルス計測）
	UART	2チャネル，16バイトFIFO，ボーレート・ジェネレータ
	USB 2.0フル・スピード	USBファンクション，12Mbps
	LCDタッチパネル・コントローラ（LTC）	抵抗膜方式タッチパネルを直結して，座標値を直読可能
	PS/2インターフェース	マウス/キーボード・インターフェース
	GPIO	汎用入出力端子×44本（ほかの機能とマルチプレクス）
	リアルタイム・クロック（RTC）	カレンダ時計，バッテリ・バックアップ可能
電　源	I/Oバッファ電源/コア電源	3.3V/1.8V
製造プロセス	ウェハ製造プロセス	0.18 μm CMOSプロセス，金属5層配線
	パッケージ	176ピンLQFP，0.5mmピッチ，外形寸法は24mm角

ことを考えて，比較的性能の高いARM9を選定しました．ARM9系のCPUの代表としてARM926とARM946があります．ARM926はMMU（memory management unit）が入っており，ARM946はMPU（memory protection unit）が入っています．Linuxなどの仮想記憶対応OSを実行することを考え，ARM926を採用しておくほうが有利であると筆者らは考えました．ただし，TLB（translation look-aside buffer）の面積分だけ，ARM926のほうがチップ面積は大きくなります．

● ビデオ表示制御や描画処理高速化のブロックを搭載

本LSIの仕様を**表1**に，内部ブロック図を**図1**に示します．

テレビやLCDへの表示をコントロールするのがVDC（ビデオ表示プロセッサ）です．VDCは各種解像度の表示に対応し，かつスクロール機能や，テレビとLCDの同時表示

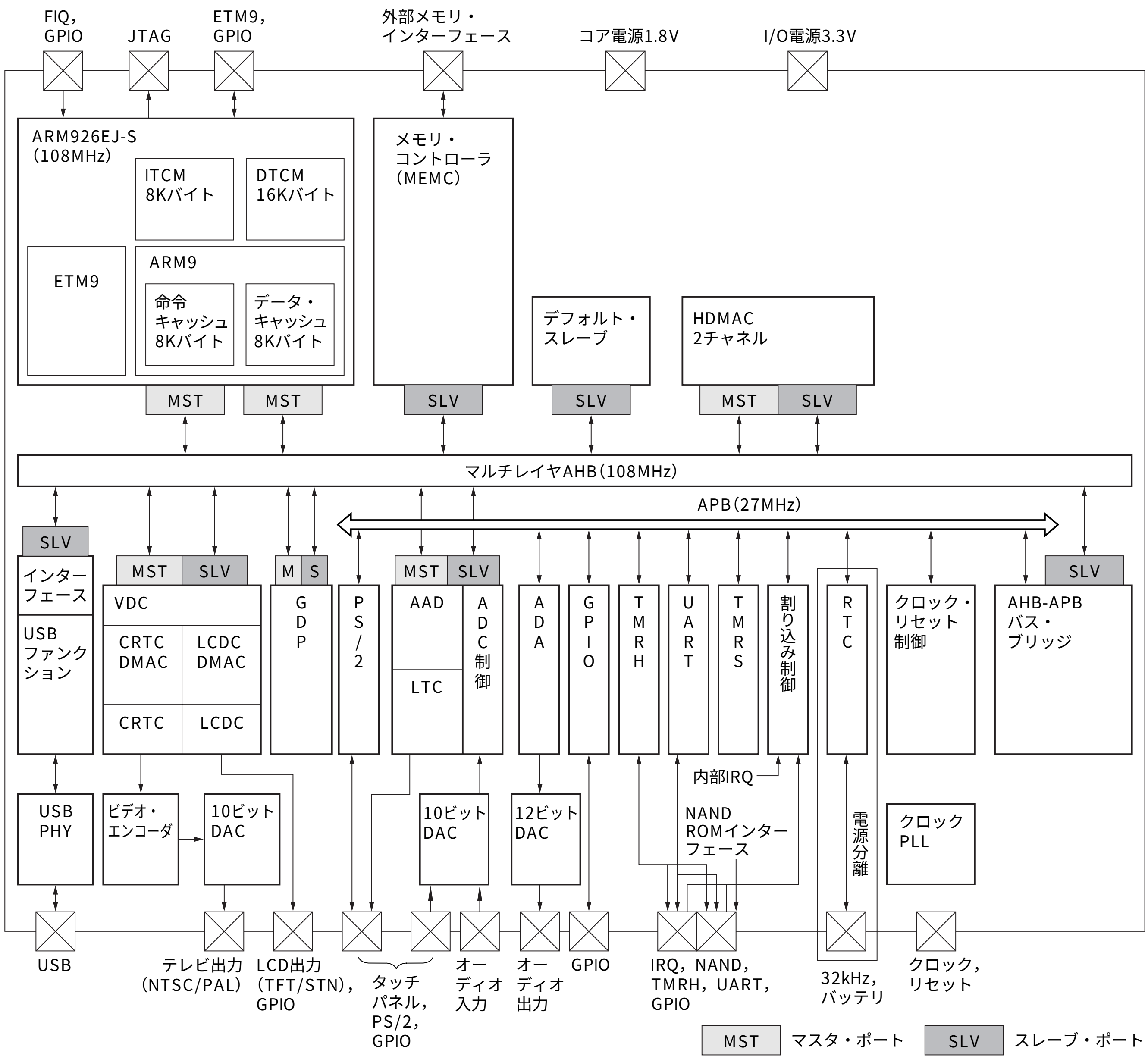


図1　開発したグラフィックスLSIのブロック図
ARM926EJ-Sをコアにして，多くの周辺機能を搭載した．中央にマルチレイヤAHBがメイン・バスとして走っている．グラフィックス性能を確保するため，CPUもAHBも同一周波数（108MHz）で設計した．このバスに，バス・マスタが6個，スレーブが8個接続されている．低速なスレーブはAHB-APBバス・ブリッジ経由でAPBに接続されている．

などを実現します．テレビとLCDに異なる絵を同時に表示できるので，例えばメイン表示をテレビに出しながら，小型液晶パネルを操作コントローラとして使う，といったアプリケーションに対応できます．

グラフィックス描画はCPUで行うことが多いのですが，アニメーション的に同じオブジェクトを少しずつ動かしながら表示する場合に，毎回CPUが描画し直しているとシステム性能の面で厳しくなります．このため，CPUの描画処理をサポートするアクセラレータGDP（グラフィックス描画プロセッサ）を内蔵することにしました．GDPは回転や拡大・縮小などをハードウェアで実現し，描画性能を大幅に向上させます．

また，抵抗膜方式の液晶タッチパネルを直結できるコントローラを入れました．LCDパネルを持つ端末類は，ほとんどタッチパネルを持っているからです．この機能としては，タッチやリリースしたことを検知し，またタッチしている座標値を直接読み取ることが可能です．

これ以外に，サウンド入出力，USBインターフェース，

UART，タイマ，DMAコントローラ，PS/2インターフェース（キーボード，マウス用），RTC（real time clock）などの周辺機能を内蔵しています．

● CPU内のメモリ容量は性能とコストから慎重に判断

ARM926EJ-Sは，ARMのV5TEJアーキテクチャに準拠したCPUで，論理合成可能なコアです．携帯電話向けのASICなどに多く採用されています．**表2**にその概要を示します．

ARM926コアは，MMUのほかに，命令キャッシュとデータ・キャッシュを内蔵しています．また，高速にアクセスできる密結合メモリ（TCM：Tightly Coupled Memory）が命令用（ITCM）とデータ用（DTCM）のそれぞれに用意されています．内蔵するこれらのメモリは，メモリ・コンパイラを使って簡単に構築できますが，そのサイズを決めるには，必要な処理性能と許されるコストを考慮した総合的な判断が必要です．本LSIの場合，仕様設計の段階でターゲットとするアプリケーション・ソフトウェアがほぼ完成していたので，必要なキャッシュやTCMの容量は早期に判断できましたが，チップ・サイズ（コスト）に大きく影響するので，システム性能とのトレードオフを考えながら慎重に判断する必要がありました．

CPUコア関係でLSI仕様に影響する点の一つとして，デバッグ用インターフェースETM（Embedded Trace Macro cell）9があります．ETM9はトレースやブレークポイントなど，ユーザ向けのデバッグ機能を提供するブロックで，**表3**に示す外部ポートが必要です．

このうち，JTAG関係の端子はLSIテストにも必要なので省けないと思います．しかし，トレース用信号としてTRACECLK以下の9〜21本のピンが必要になり，コストを抑えるためにLSIパッケージのピン数に制限があるときは，この本数がネックになります．本LSIでは，TRACEPKT信号を4本とし，合計9本のトレース信号を出しています．同時にこれらの端子にGPIO機能をマルチプレクスし，専用の外部信号1本で機能を切り替えられるようにしています．なお，ETM9はベンダからIPコアとして提供してもらう際に，内蔵するデバッグ機能の規模を3〜4通りの中から選択することになっています．

● CPUまわりの仕様判断はソフトウェア技術者とも相談

また，ARM926のコアそのものには，CPUの初期動作を決めるための下記の入力信号があります．

- **VINITHI** ―― リセット直後の例外ベクタ領域のアドレスを決める信号．0の場合は，リセット直後の例外ベクタ領域が0x00000000〜0x0000001C．1の場合は0xFFFF0000〜0xFFFF001C．
- **BIGENDINIT** ―― リセット直後のエンディアンを決める信号．0の場合は，リセット直後はリトル・エンディアン．1の場合はビッグ・エンディアン．
- **INITRAM** ―― リセット直後のITCM有効状態を決める信号．0の場合は，リセット直後，ITCMは無効．1の場合は有効．

これら各信号は，各種CPU動作を設定するシステム制御コプロセッサ（CP15）レジスタの対応するビットの初期

表2　ARM926EJ-Sの概要

項　目	内　容
アーキテクチャ	● ARMアーキテクチャ V5TEJ準拠 - 16ビット長（Thumb）命令サポート - 信号処理命令セット・サポート - Javaバイト・コード実行サポート
ハーバード・バス	● 命令バスとデータ・バスの分離
TCM	● 密結合メモリ（高速内蔵RAM） - 命令用密結合メモリ：ITCM（4K〜1Mバイト） - データ用密結合メモリ：DTCM（4K〜1Mバイト）
キャッシュ	● 4ウェイ・セット・アソシアティブ ● 32バイト/ライン - 命令キャッシュ：4K〜128Kバイト - データ・キャッシュ：4K〜128Kバイト
MMU	● アドレス変換 ● データ・キャッシュ設定
ライト・バッファ	● ライト・レイテンシの低減
ETM9	● トレース機能，デバッグ機能のサポート

表3　ARM9デバッグ・ポート

項　目	信号名	方　向	内　容
TAP インターフェース（JTAG）	TCK	入力	JTAGテスト・クロック
	TMS	入力	JTAGテスト・モード
	nTRST	入力	JTAGテスト・リセット
	TDI	入力	JTAGテスト・データ入力
	TDO	出力	JTAGテスト・データ出力
	RTCK	出力	JTAGリターン・クロック注1
	nSRST	入出力	JTAGシステム・リセット注2
TRACE	TRACECLK	出力	トレース・クロック注3
	TRACESYNC	出力	トレース同期化信号
	PIPESTAT [2：0]	出力	ARMパイプライン・ステータス
	TRACEPKT [*N*-1：0]	出力	トレース・パケット注4

注1：ICEがTDOを取り込むクロック
注2：オープン・ドレイン出力
注3：フル・レート，ハーフ・レートの選択あり
注4：バス幅は4，8，16ビットから選択

値を決める信号です(稿末のコラム「ARM926コアの動作設定」を参照).

このうちVINITHIとBIGENDINITについては，リセット・ブート時に選択できる意味はほとんどなく，LSI内で固定にするケースが多いと思います．INITRAMについては，固定にする場合は「リセット後，無効」にするか，あるいはITCMにブート・コードをロードしたあとでリセット後のブートをITCM内の命令で行ってもかまわないので，LSI内に置くレジスタでINITRAMを制御するケースもあります(そのレジスタ値は，コールド・リセットでゼロ初期化)．こうしたCPUまわりの仕様は，LSI設計者とソフトウェア開発者が協議して判断しなくてはなりません．

● 改良版のマルチレイヤAHBを採用

ARMコアをベースにLSIを設計するとなると，そのバス・アーキテクチャであるAMBA(Advanced Microcontroller Bus Architecture)をよく知る必要があります．AMBAはいくつかあるバス方式の総称であり，もっとも普及しているのがAMBA Rev.2です．このうちの代表的なものにAHB(Advanced High-performance Bus)とAPB(Advanced Peripheral Bus)があります．

- **AHB** ――高速転送用のバスで，アドレス・フェーズとデータ・フェーズがずれた，いわゆるパイプライン型のバス．コア側クロック(HCLK)の1サイクル・ピッチで転送できる．
- **APB** ――周辺モジュールを接続する低速バスで，アドレスとデータは同一フェーズに現れる．周辺側クロック(PCLK)の2サイクル・ピッチで転送する．AHBとAPBをインターフェースするブリッジ回路がIPベンダから提供されている．

最近のシステムLSIでは，純粋なAHBバスではなく，その改良版のマルチレイヤAHBを採用することが多くなっています．本LSIでもこれを採用しています．その考えかたを**図2**に示します．マルチマスタとマルチスレーブを簡単に接続でき，複数マスタのアクセス先スレーブが異なっていれば並行して同時アクセス可能です．これにより，システム性能を向上させることができます．各モジュールとマルチレイヤAHBの間は，AHB仕様からアービトレーション関係の信号を省いた「AHB Lite」というバス規格で接続します(**図3**，**表4**)．

図3で動作を説明します．マスタはスレーブ側(バス側)から返るHREADYOUT信号がHIGHのときに内部ステートを進ませ，アドレスHADDR①を出力します．次のHREADYOUT＝HIGHで次アドレスHADDR②を出力します．このHADDR②を出力しているときが，①のデータ処理(リード/ライト)タイミングになります．ライト時はこのタイミングでデータHWDATA①をマスタが出力し，リード時はデータHRDATA①がスレーブ側(バス側)から返されます．このようなパイプライン動作になっているので，スレーブ側がデータを処理する時間の余裕が生まれ，クロック周波数を上げることができます．スレーブ側のデータ処理が間に合わない場合は，スレーブ(バス)側はHREADYOUTをネゲートします．**図3**では，②のデータ・フェーズがHREADYOUTにより延ばされています(ウェイト処理)．

従来のAHBバスでは，マスタはバス・アービトレーション制御を考慮する必要がありましたが，ここではその必要はありません．基本的にバスから返るHREADYOUT信号を見て内部ステートを進ませればよいようになっており，設計の見通しが良くなっています．

● バス内のマスタの優先順位は固定

ところで**表4**を見ると，レディ信号として，HREADYとHREADYOUTの2本があることがわかります．スレー

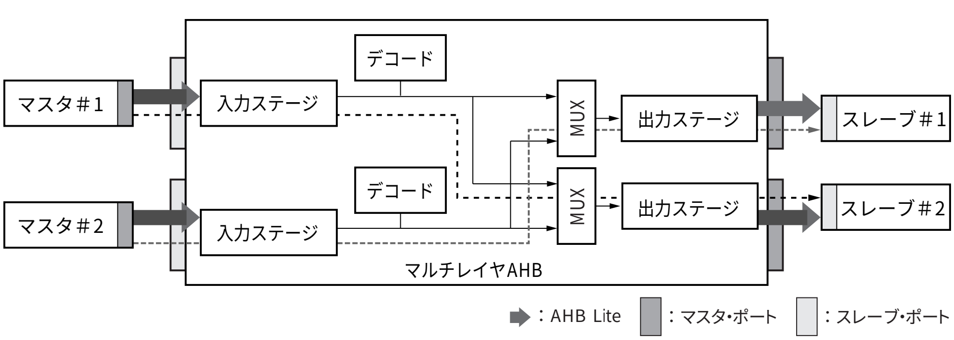
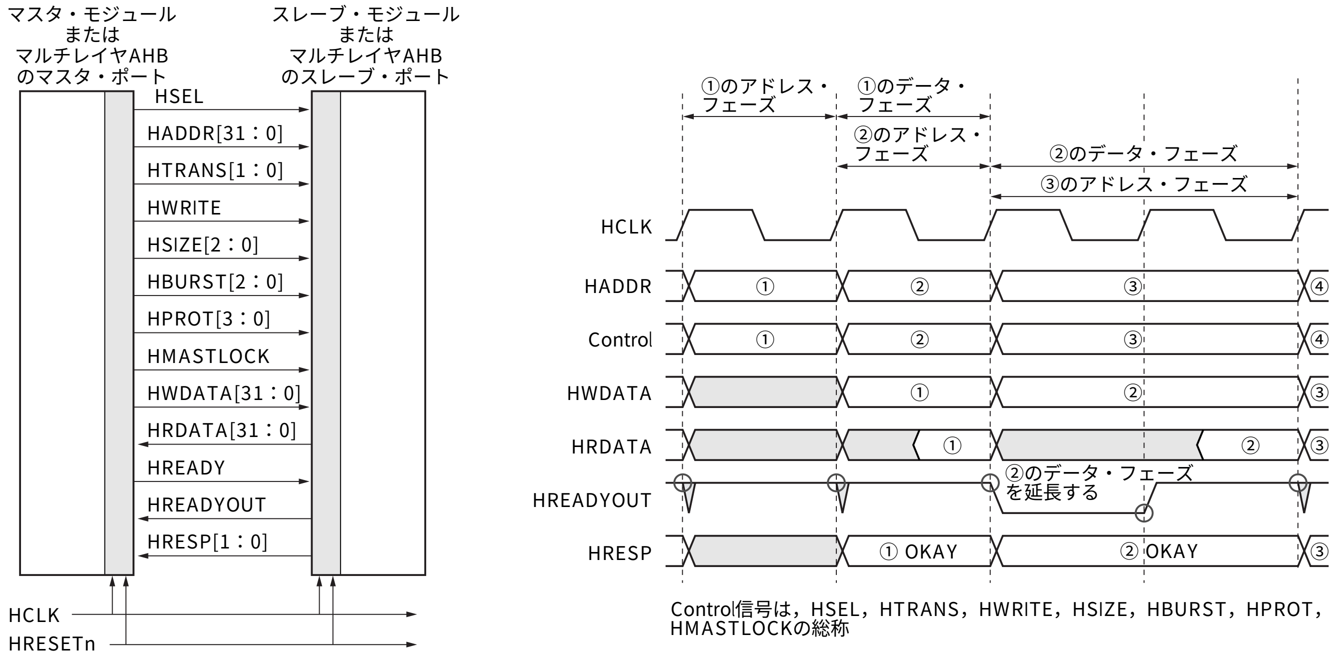


図2 マルチレイヤAHB
マルチレイヤAHBは，複数のマスタ・モジュールと，複数のスレーブ・モジュールを相互結線する構造を持つ．例えばこの図では，マスタ#1がスレーブ#2にアクセスするのと同時に，マスタ#2がスレーブ#1にアクセスできる．各モジュールとマルチレイヤAHBの間は，AHB-Lite規格のバス信号で結ばれている．



（a）マスタとスレーブの接続　　（b）タイミング・チャート

図3　AHB Lite

AHB Liteは，標準的なAHBバスからアービトレーション関係の信号を省いたものである．マルチレイヤAHB上では，複数のバス・マスタが存在していてもバス側で調停を行い，マスタ側は基本的にはHREADYOUTだけを見てバス生成の制御を行う．これにより，設計上の見通しが良くなる．

ブ側でそのデータ処理が間に合わないとき，バスのデータ・フェーズを延ばすために出力するのがHREADYOUTです．一方，HREADYはスレーブへの入力信号になっています．これはスレーブが前サイクルで選択されていたほかのスレーブへのアクセスが終了するタイミングを知り，次の自分の転送データの準備を行うために使います．各スレーブからのHREADYOUTはマルチプレクサでまとめられ，グローバルHREADYとして全マスタに伝えられるとともに，全スレーブへのHREADY入力になります．多くのケースでは，スレーブがデータ・フェーズを延ばすためのHREADYOUTだけを用いるようです．本LSIでもHREADYOUTしか使っていません．

本LSIのマルチレイヤAHBには，**図1**に示したように，CPUの命令ポートとデータ・ポートを含めてマスタが6個接続されています．各マスタは勝手にアクセスを発生させるので，同じスレーブに対するアクセスが競合する場合は，このバス内で優先順位の調停が必要です．マルチレイヤAHB上のマスタの優先順位は，「ラウンドロビン方式」と「固定方式」を設計時に選択します．本LSIでは，VDCによるグラフィックス表示のためのフレーム・メモリのアクセスが最優先なので，以下のように固定にしました．

VDC＞HDMAC＞GDP＞AAD（DMA機能）
＞ARM926データ＞ARM926命令

さて，本LSIの設計において，このバスで苦労した点があります．本LSIはグラフィックス処理チップなので，フレーム・メモリが置かれるSDRAMをアクセスするバス帯域がシステム性能を左右します．このため，CPUの動作周波数とマルチレイヤAHBの動作周波数を同一かつ最大にしたいと考えました．テレビ表示を考えているので，選択できる周波数はビデオ・クロック27MHzの整数倍（81MHz，108MHz，135MHz，…）に限定されます．本LSIが採用する0.18 μmプロセスでは，ARM926コアは余裕で108MHz動作しますが，一般のARMのシステムLSIでは，AHBバスの動作周波数はCPUの半分にするケースが多いようで，バスはタイミング的に108MHzで動かせる特性を持っていませんでした．かなり苦労しましたが，各モジュールの論理段数の削減やレイアウトのくふうにより，バスも108MHzで動作させるようにしました．

● 2面のフレーム・メモリでピンポン・バッファを構成

テレビやLCDパネルにグラフィックスを表示する場合，

表4　AHB Lite信号の意味

信号名	出力元	意　味
HSEL	マスタ	スレーブ・セレクト
HADDR[31：0]	マスタ	アドレス
HTRANS[1：0]	マスタ	転送タイプ 00 IDLE　アクセスなし 01 BUSY　バースト・アクセス中の一時停止状態で，スレーブはバス再開に備える必要がある 10 NOSEQ　不連続アドレスの個別アクセス 11 SEQ　連続アドレスのバースト・アクセス
HWRITE	マスタ	ライト 0 リード動作 1 ライト動作
HSIZE[2：0]	マスタ	アクセス・サイズ 000 8ビット　100 128ビット 001 16ビット　101 256ビット 010 32ビット　110 512ビット 011 64ビット　111 1024ビット このアクセス・サイズ情報は，WRAPバースト・アクセスのアドレス境界を判断する際にも使われる
HBURST[2：0]	マスタ	バースト・アクセス方法 000 SINGLE　100 WRAP8 001 INCR　101 INCR8 010 WRAP4　110 WRAP16 011 INCR4　111 INCR16 SINGLEは非バースト・アクセスを示す． INCRは任意長のアドレス・インクリメント・バースト INCR*n*バーストは，*n*回のアドレス・インクリメント・バースト WRAP*n*バーストは，アドレスが回り込む （例：0x34からのWRAP4は，0x34→0x38→0x3C→0x30） なお，INCR，INCR*n*，WRAR*n*のバースト・アクセスでは1Kバイト境界をまたぐことはできない
HPROT[3：0]	マスタ	プロテクション制御 HPROT[3]　0＝非キャッシャブル，1＝キャッシャブル HPROT[2]　0＝非バッファラブル，1＝バッファラブル HPROT[1]　0＝ユーザ・アクセス，1＝特権アクセス HPROT[0]　0＝OPコード・フェッチ，1＝データ・アクセス
HMASTLOCK	マスタ	このマスタがバスを取り続けたい場合にアサート
HWDATA[31：0]	マスタ	ライト・データ
HRDATA[31：0]	スレーブ	リード・データ
HREADY	マスタ	レディ入力：スレーブへの入力信号． 当該スレーブが，ほかのスレーブへのアクセスが終了するタイミングを知るときに使う
HREADYOUT	スレーブ	レディ出力：スレーブからの出力信号． そのアクセスに時間がかかるとき，マスタにデータ・フェーズを延ばしてもらうときに使う
HRESP[1：0]	スレーブ	レスポンス 00 OKAY　アクセス成功 01 ERROR　アクセス・エラー発生 10 RETRY　マスタへの連続リトライ要求（当該バス・マスタの優先順位は一定のままでリトライさせる） 11 SPLIT　マスタへの間欠リトライ要求 （当該バス・マスタの優先順位を一時的に下げて，リトライに間が空いてもよい）

フレーム・メモリ上に絵を描画（ライト）してそれを表示タイミングに合わせて横方向に走査（リード）し，ピクセル値を順次テレビやLCDパネルに送ります．このとき，表示中（走査中）のフレーム面に非同期に描画すると，チラツキが見えることがあります．

これを避ける手として，表示走査を行わない垂直ブランキング期間に描画する方法があります．この方法はフレーム・メモリを1面だけ持てばよいので，メモリ容量が少ない場合には有効です．しかし，垂直ブランキング期間は限られているので，複雑な描画を完了させられないケースが生じます．

このため本LSIでは，**図4**に示したようにフレーム・メモリを2面持ち，ピンポン・バッファを構成することを前提としました．片方のフレーム・メモリを表示しているときは，もう一方に描画するようにします．描画期間として1フレーム周期（30フレーム/sのアニメーションの場合は33ms）まるまる確保できるので，複雑な描画処理も行えます．これにより，スムーズなアニメーション（動画）表示が可能となりますが，メモリ容量は多くなります．ただし，16ビット幅のSDRAMは単体ですでに容量が大きく，入手できる最小容量のものでも16Mビット＝2Mバイトあるので，VGA（640×480）サイズのフレーム・メモリ2面分は

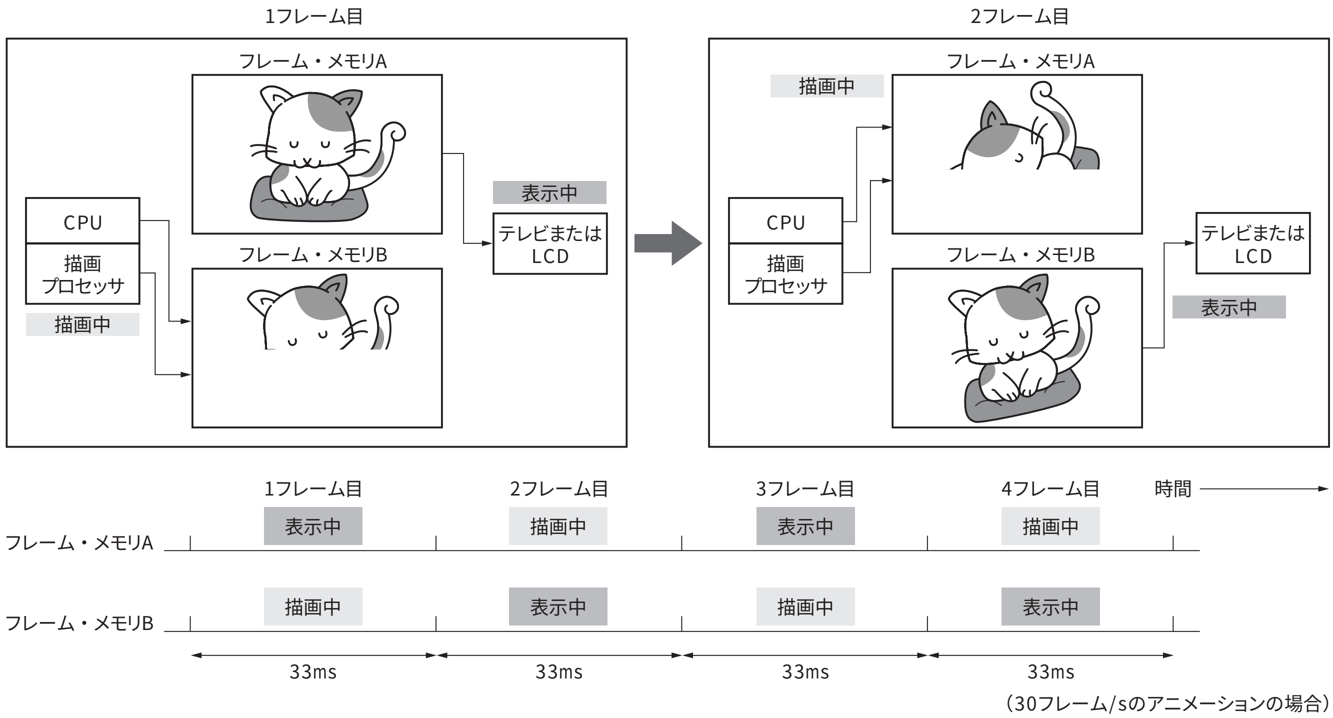


図4　グラフィックスの描画と表示

本LSIでは，テレビやLCDパネルに動画などのグラフィックスを表示する場合，内部処理を大きく「描画」と「表示」の二つに分けている．まず，CPUや専用プロセッサがグラフィックスをフレーム・メモリ上にライトする．これを「描画」という．次に，描画されたグラフィックス・データをフレーム・メモリの走査（リード）を行いながらテレビやLCDパネルに送る．これを「表示」という．あるフレーム・メモリを表示しているときに，同じフレーム・メモリに描画すると，画面上にチラツキが見えてしまう点に注意が必要である．このため，一般にフレーム・メモリを2面に分けて，片方を描画中のときはもう一方を表示している．そして，テレビやLCDの垂直周期で切り替える．このようなフレーム・メモリ方式をピンポン・バッファという．

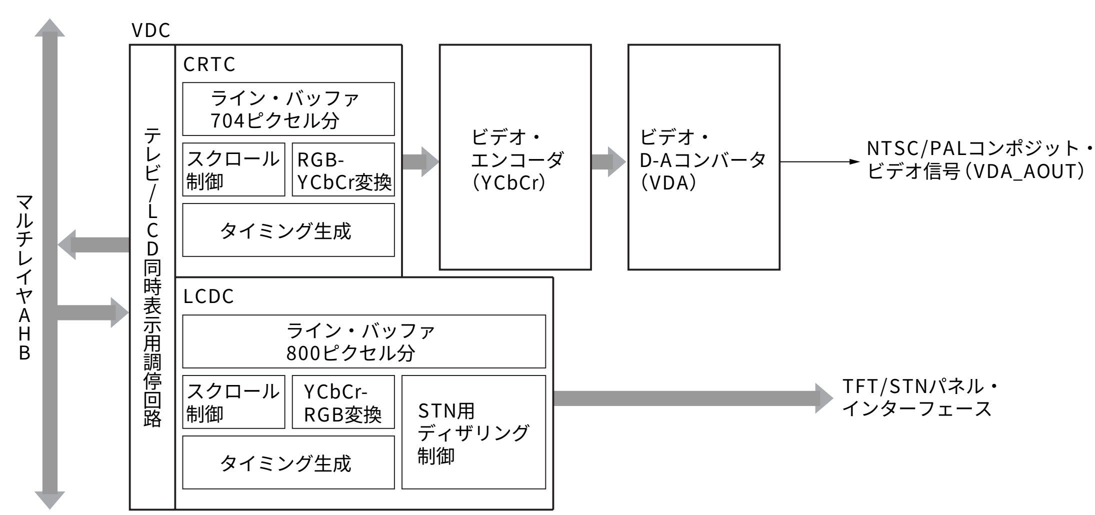


図5　VDCのブロック図

VDCは，テレビ表示用のCRTCとLCD表示用のLCDCから構成される．それぞれがフレーム・メモリを走査するためのバス・マスタになっており，CRTCはテレビのVGA表示時の横方向ピクセル数分の，LCDCはSVGA表示時の横方向ピクセル分のライン・バッファ（FIFOメモリ）を持っている．水平方向のブランキング期間の開始時からフレーム・メモリを横方向に走査（リード）し，ライン・バッファに格納する．ライン・バッファからは，テレビあるいはLCDの表示タイミングに合わせてピクセル値を取り出し，それぞれの表示装置に送る．マルチレイヤAHBのインターフェースのブロックには，テレビとLCDを同時に表示するための調停回路が入っている．CRTC側のビデオ・エンコーダIPはYCbCrデータでインターフェースする仕様だったので，CRTC内でRGBからYCbCrへの変換を行った．フレーム・メモリ上のピクセル値をYCbCrにするモードに設定されているときは，この変換回路をバイパスしている．LCDC側は基本的にはRGB値をそのままパネルに送出すればよいが，フレーム・メモリ上のピクセル値をYCbCrにするモードに設定されているときはYCbCrからRGBに戻す変換回路を通す．

確保できます．

● LCD表示は変則的な解像度にも対応

VDCは，グラフィックス制御のうちの表示動作を受け持ちます．フレーム・メモリ上に描画された絵を，横方向に順次走査して，テレビやLCDに表示タイミングに合わせてピクセル値を送り出す機能を持ちます．

テレビ表示の解像度は，NTSC（National Television System Committee）モードではVGA設定で704×480，QVGA設定で352×240です．PAL（phase alternation by line）モードではVGAで704×480または704×576，QVGAで352×240または352×288です．QVGAでは，奇数本目の走査線と偶数本目の走査線に同じピクセル値を表示するので，ノン・インターレース的になります．とくに，VGA解像度（インターレース）で横方向の細い線を表示すると，テレビの原理上チラツキが目立ちます．そういう表示を嫌うアプリケーションでは，QVGAに設定したほうがよい場合があります．

LCD表示の解像度は，TFTパネルではQVGA（320×240），VGA（640×480），SVGA（800×600）に対応しています．このクラスのLCDパネルであれば，VDCの設定により，ワイド・タイプなどの変則的な解像度にも対応できます．STNパネルについてはQVGA（320×240）以下のサイズに対応しています．

VDCの内部は**図5**のように，テレビ用のCRTC（CRTコントローラ）とLCDパネル用のLCDC（LCDコントローラ）に分かれています．CRTCの出力はビデオ・エンコーダとビデオDAC（D-Aコンバータ）を通して，コンポジット・ビデオ信号としてLSIから出力します．LCDCの出力はLCDパネルに直結可能な信号として出力しています．TFTに対してはRGBパラレル・データと同期信号，STNに対しては8ビット・データと同期信号です．

CRTCとLCDCのそれぞれが横方向1ライン分のライン・バッファを持っています．1ラインの走査（SDRAM上のフレーム・メモリのリード）を開始したら，一気にバースト読み出しを行ってピクセル値をライン・バッファにためます．テレビやLCDパネルに対しては，表示タイミングに合わせて順次，ライン・バッファからピクセル・データを送出していきます．

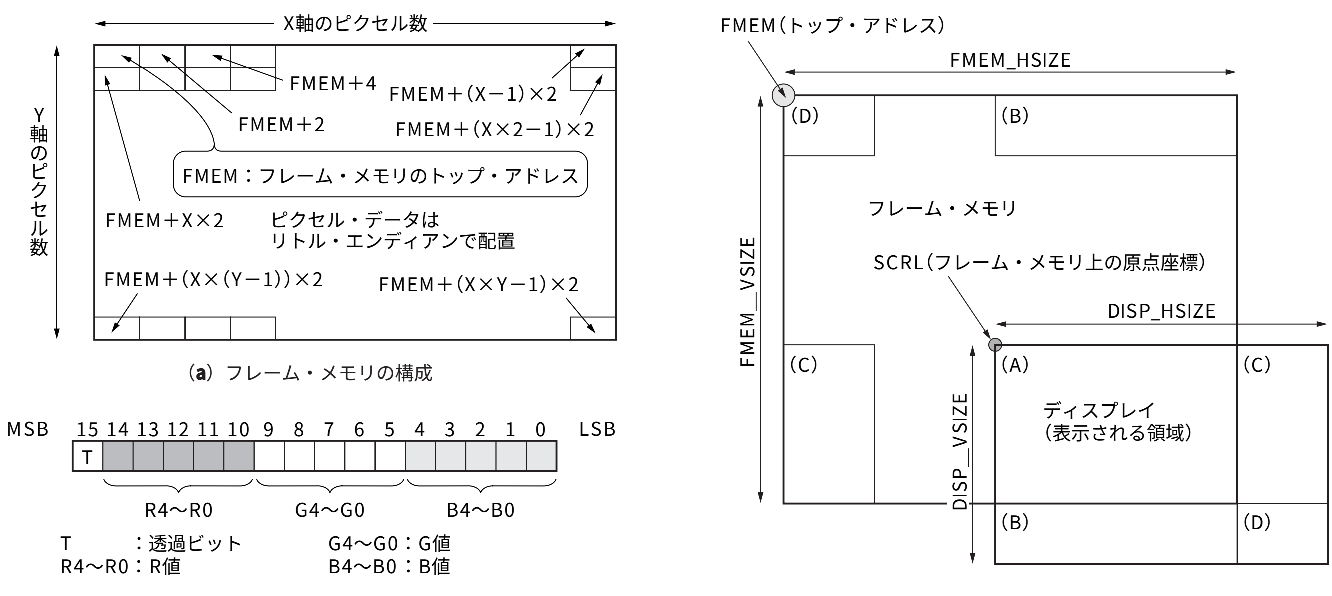


図6　フレーム・メモリの構成

フレーム・メモリは，通常，外部のSDRAM上にアロケートする．(a)に1面分のフレーム・メモリの構成を示す．画面横方向にアドレスがインクリメントするようにピクセル値が配置される．1ピクセルは(b)に示したような16ビット・フォーマットである．RGBそれぞれ5ビットずつで，32768色を表現する．ピクセル値をRGBで持たせるモードのほかに，YCbCr値で持たせるモードもあり，動画再生に対応している．最上位ビットの透過ビットTは，GDPが使うビットである．フレーム・メモリ上にグラフィックス・データを転送するとき，Tビットが1のピクセル値は転送しないことで透明化処理を行っている．なお，GDPは，特定の色を透明と判断してピクセル値をフレーム・メモリに転送しないこともできる．フレーム・メモリは，最大4094ピクセル×4095ピクセルのサイズまで確保できる．フレーム・メモリ上でテレビやLCDパネルが実際に表示する位置を任意に指定でき，これによりハードウェア・スクロールを実現している．(c)のように，はみ出た部分は上下左右折り返して表示でき，背景の連続スクロールなどの表現も可能となっている．

● テレビとLCDを同時に表示する

図6にフレーム・メモリの構成を示します．1ピクセルは16ビットで，32768色（RGB 555）を表現します．これらが画面横方向に向けて順次，アドレスがインクリメントするように連続して格納されます．フレーム・メモリは実際の表示サイズ（VGAなど）よりも大きくとることができ，表示領域の原点を変更してスクロール表示させることができます．フレーム・メモリの先頭番地は，1垂直周期ごとに変更できるため，これによりピンポン・バッファを実現しています．先頭番地は任意の位置を指定できるので，原理的には2面を超えるフレーム・メモリにも対応できます．描画処理が重くなる場合に3面ないしは4面に増やす，すなわち描画処理時間のジッタを吸収するリング・バッファ段数を増やした形にして，グラフィックス性能を確保することも可能です．

図7に，テレビやTFTパネルの場合の表示タイミングを示しました．**図7（a）**のように水平ブランキング期間が始まったら，フレーム・メモリの水平方向1ラインを一気にバースト・リードします．またVDCは，テレビとLCDの同時表示が可能です．**図7（b）**のようにテレビとTFTは水平周期が非同期の関係になりますが，アクセスが競合するタイミングでは最大128ピクセルごとに切り替えて，表示動作が破たんしないようにしています．

CRTCとLCDCからは，独立に垂直周期で割り込みが出力されており，アプリケーション側でその周期に同期して描画処理とフレーム・メモリの先頭アドレスの切り替えを行います．テレビとLCDの同時表示の場合は，テレビ用とLCD用に二つのタスクを用意して，テレビ用タスクはCRTCからの割り込みで，LCD用タスクはLCDCからの割り込みで，それぞれ描画とフレーム・メモリ切り替えを

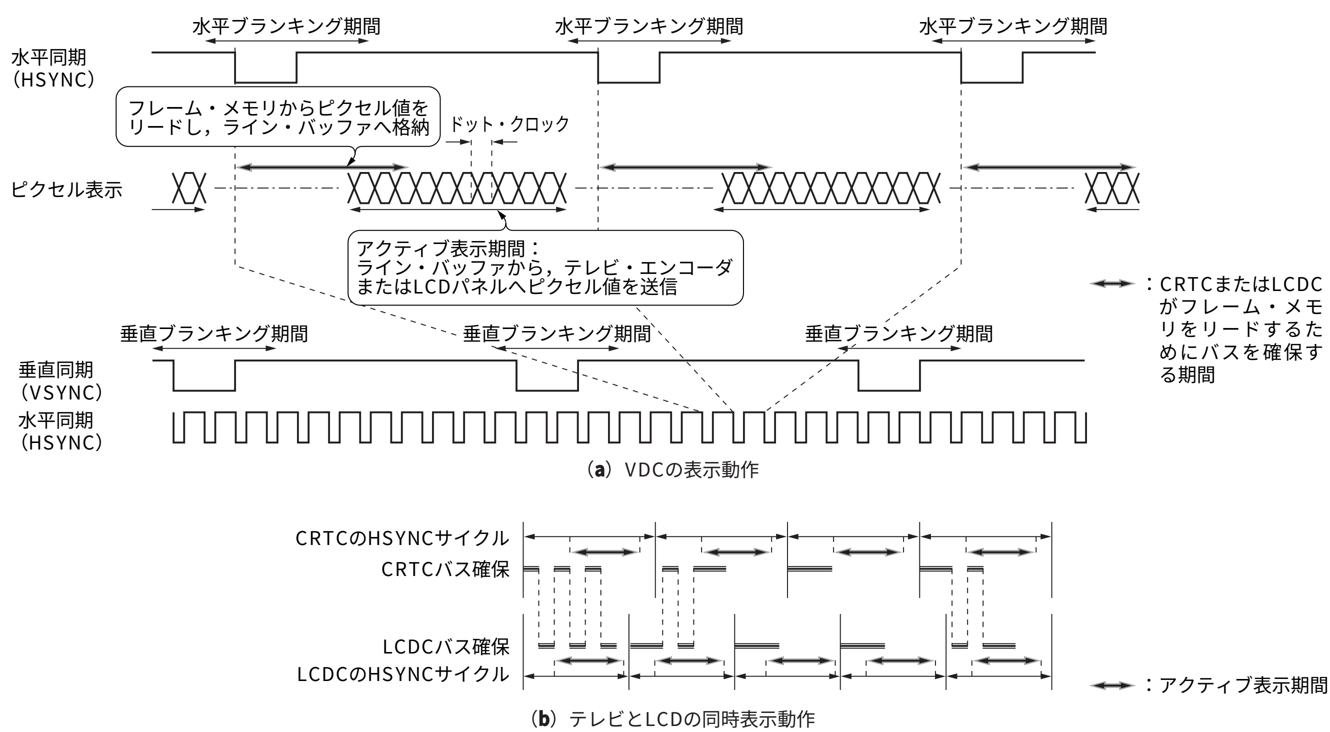


図7　VDCの表示動作

テレビやTFTパネルの機能的なタイミング動作は基本的に同じで，（a）に示したようになっている．垂直同期（VSYNC）タイミングがあり，その1周期中に走査線本数に応じた水平同期（HSYNC）タイミングがある．TFTパネルの垂直同期は約60Hzであり，この1周期で1面分の表示を行う．テレビ（NTSC，PAL）はインターレース走査なので，垂直同期は60Hzで発生するが，2×VSYNCの期間（30Hz）で1面分の表示を行う（飛び越し走査に対応したEVEN/ODD表示）．垂直方向にも水平方向にも表示を行わないブランキング期間がある．表示を行う水平位置の水平ブランキング期間の開始と同時に，フレーム・メモリを1水平ライン分読み込んで，ライン・バッファへ表示ピクセルを送る．その後，内部表示タイミングに従って，テレビの場合はテレビ・エンコーダへ，TFTの場合は直接パネルへ，ピクセル値を出力していく．一方，ここでは図示しないが，STNパネルの場合はブランキング期間がないもののQVGAクラスだとドット・クロックが遅いので，1ラインの表示が終わった直後に次ラインのフレーム・メモリ読み出しを行っても十分に間に合う．（b）のように，VDCはテレビとLCDの同時表示が可能である．テレビとLCDは解像度が異なるため，水平・垂直周期はお互いに非同期になる．このため，非同期にフレーム・メモリをアクセスすることになるがCRTCとLCDCのフレーム・メモリ・アクセスが競合した場合は，最大128ピクセル（256バイト）をリードするたびに，CRTCとLCDCのバス権を交代させるような調停制御を行っている．

独立に実行します．これによって，異なるアニメーション（動画）を表示させることができるわけです．

LCDパネルのうち，STNパネルは各ピクセルがON状態かOFF状態しかないので，階調表示させるには，チラツキやビートを抑制するための特殊変調したパルスによるディザリング処理が必要です．LCDCにはこのための制御回路があり，STNパネルについても32768色の発色を可能にしています．

● 描画動作は矩形領域を別の位置に移すBitBltが基本

GDPは，グラフィックス制御のうちの描画動作を受け持ちます．描画処理はフレーム・メモリへの絵の書き込みの作業であり，CPUとGDPが互いに協調して実行していきます．CPUはおもに材料作りを受け持ちます．JPEG画像の展開やベクトル図形（直線，円弧など）の作成を行い，ワーク・メモリ上に準備します．GDPは，この完成した材料をフレーム・メモリ上の指定された座標に転送します．材料は1度作成すると，何回でも使い回しができます．一方，フレーム・メモリにアニメーション（動画）として描画する作業は毎フレーム必要です．ここでGDPの性能が生きてきます．

GDPの転送の基本は，**図8（a）**に示したBitBlt（ビットブリット）（bit block transfer）です．これは，ある矩形領域の図形を別の位置に転送するものです．**図8（a）**はまったくそのままのサイズで転送していますが，拡大・縮小したり回転したりしながら転送することも，グラフィックスのコンテンツでは頻繁に起こります．このため，GDPは**図8（b）**のようにアフィン変換処理[注]を伴うBitBlt転送をサポートしています．

アフィン変換とは，例えば，元絵を角度θだけ回転し，X方向にsx倍，Y方向にsy倍伸縮させ，かつX方向にTX，Y方向にTYだけ並行移動させる，といった図形処理です．このケースでは，元絵（ソース・ピクセル）内の各座標（$X1$, $Y1$），転送先（デスティネーション・ピクセル）の各座標を（$X2$，$Y2$）とすると，下記の関係になります．

$$\begin{pmatrix} X2 \\ Y2 \end{pmatrix} = \begin{pmatrix} SX & RY \\ RX & SY \end{pmatrix} \begin{pmatrix} X1 \\ Y1 \end{pmatrix} + \begin{pmatrix} TX \\ TY \end{pmatrix} \quad \cdots\cdots\cdots\cdots (1)$$

ただし，

$$SX = sx \times \cos\theta,\ RY = sx \times (-\sin\theta)$$
$$RX = sy \times \sin\theta,\ SY = sy \times \cos\theta$$

です．**図8（b）**のSRC_RECT内の各ピクセルが転送先のどこに位置するかをすべての点に対して演算する必要があり，CPUで処理すると非常に時間がかかります．本LSIで

注：アファイン（affine）変換はアフィン変換と呼ばれることもある．

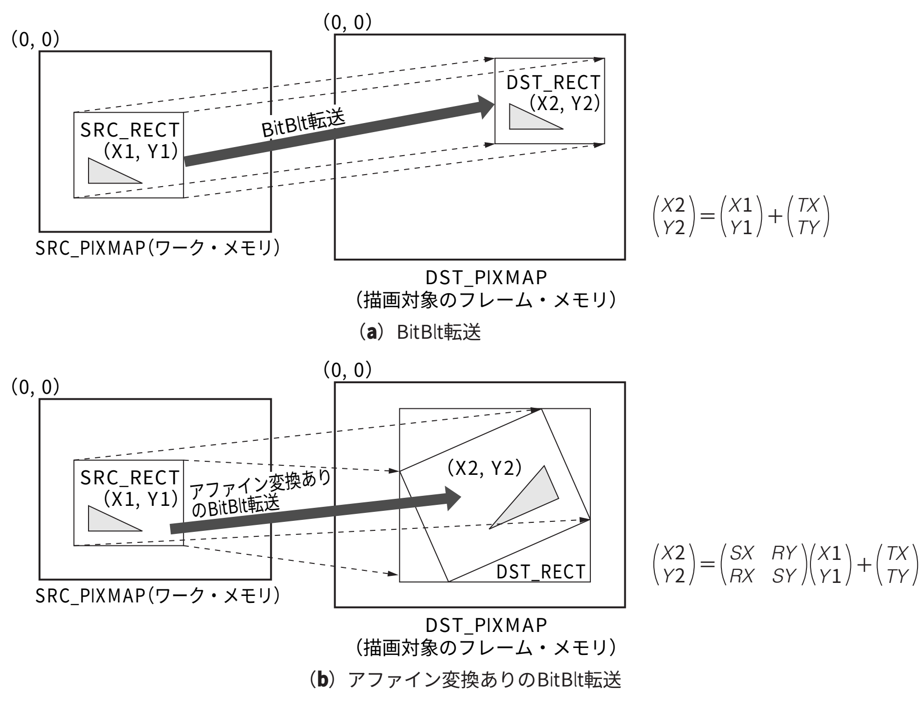


図8
BitBltとアファイン変換
（a）はシンプルなBitBlt転送である．SRC_PIXMAP中の任意位置・任意サイズの長方形の内部ピクセルを，DST_PIXMAPの任意位置に同じサイズで転送する．（b）はアファイン変換を伴うBitBlt転送である．拡大縮小と回転を行いながら，SRC_PIXMAP中の任意位置・任意サイズの長方形SRC_RECTの内部ピクセルをDST_PIXMAPに転送する．

は，これをハードウェア（GDP）で実現して性能を確保しています．

なお，アファイン変換において，ソース側のピクセル値のアファイン変換後の座標を計算してそこに転送するだけですと，拡大転送時などに転送先ピクセルがすべて埋まらず，歯抜けになることがあります．このためGDPでは，転送先を囲む最大矩形範囲内（DST_RECT）の全ピクセルに対して逆アファイン変換を施して，ソース側のどのピクセルに対応するかを見てからDST_RECT内のピクセルを埋めています．これにより歯抜けになることを防げます．

● 透明処理や半透明処理もハードウェアで実行

GDPのBitBlitにおいて，転送する絵の一部を透明処理することができます．透明にする色を指定する方法と，透明にするピクセルのMSB（Tビット）を1にする方法があります．透明処理は，透明と指定されたピクセルを転送しないだけで実現できます．

また，**図9**に示す半透明処理にも対応します．これをアルファ・ブレンディングと呼びます．透過率（アルファ値）をαとすると，

$$転送するピクセル値＝ソース・ピクセル値\times\alpha＋下地のピクセル値\times(1-\alpha)$$

で計算できます．これもハードウェア（GDP）により対応します．

さらにGDPは，転送する絵の色相・彩度・明度を変更しながら転送することができます．RGBそれぞれの色要素に対して独立に掛ける係数AとオフセットBを指定できます．

$$転送するピクセル値＝ピクセル値\times A+B$$

これにより，1個の材料にいろいろな効果を付けてフレーム・メモリ上に展開できます．

GDPの内部構造は**図10**に示すように，CPUに似た構造を持っています．アファイン変換における座標演算やアルファ・ブレンディング，色変換などの各処理は，固定小数点による高速演算回路で処理しています．

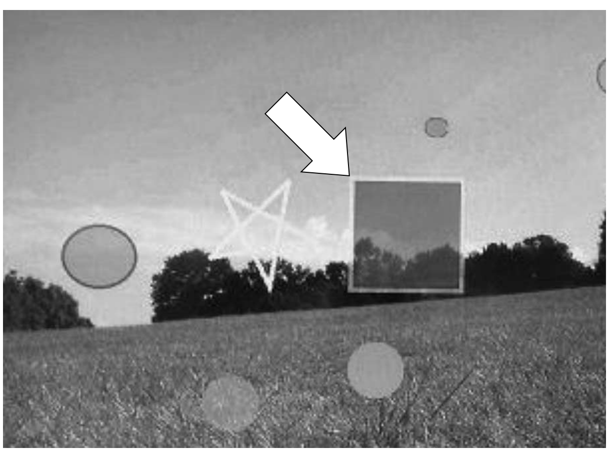
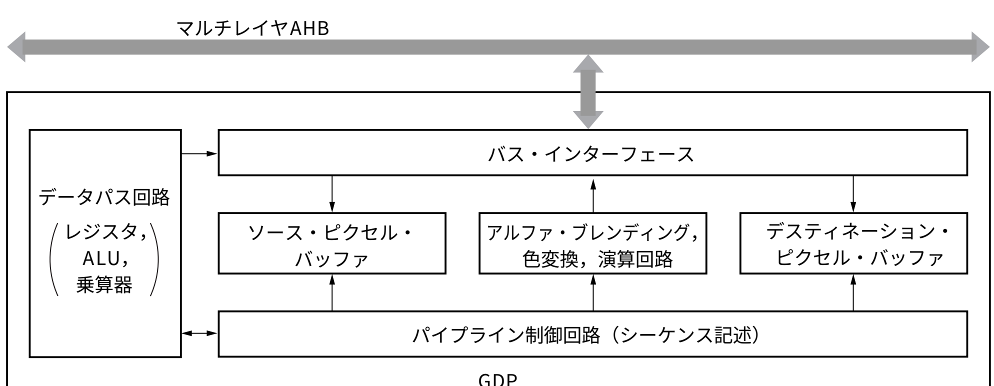


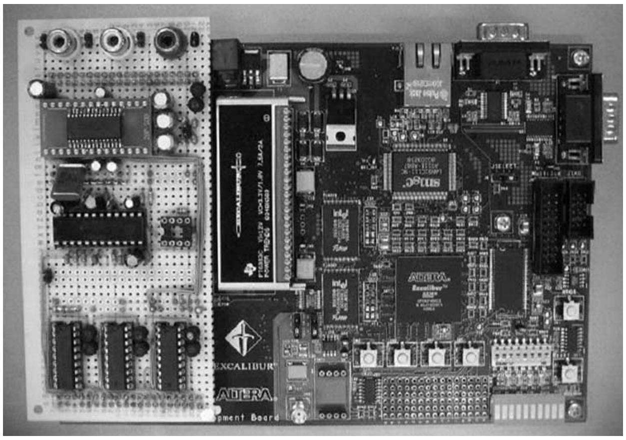
図9　アルファ・ブレンディングの効果
転送先にもともとあるピクセル値と転送するピクセル値に重みを付けて加算することで，半透明処理を実現できる．GDPは，32段階の透過率を指定できる．

● LSIの仕様を決める前に応用ソフトを開発して評価

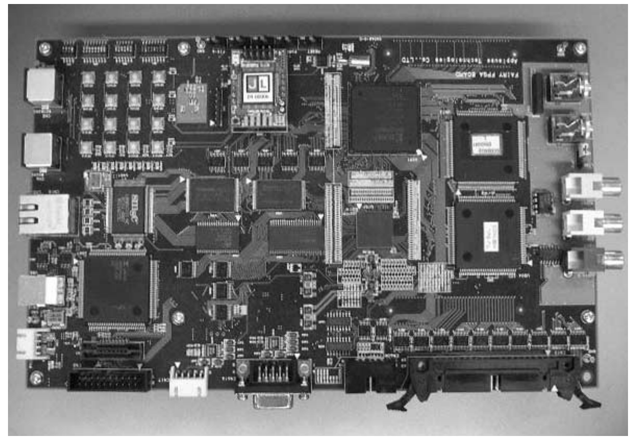
本LSIの開発プロジェクトでは，LSIの仕様を決める前にアプリケーション・プログラムの開発を始めました．ARM922Tコアを内蔵した米国Altera社のFPGAを用い，この中に簡単なテレビ表示回路とサウンド用のPCM（pulse code modulation）出力回路を構築しました（**写真1(a)**）．

図10
GDPのブロック図
GDPは，グラフィックス・データをフレーム・メモリにBitBlt転送する一種のDMAコントローラだが，アファイン変換やアルファ・ブレンディングなどの複雑な処理があるため，構造はCPUに近い．内部的には，命令に相当するシーケンス記述がワイヤード論理で記述されており，それに従って5段パイプライン制御で動作する．BitBlt転送時のSDRAM性能の効率向上のため，一度に16ピクセルを連続リード/ライトできるように，ソース・ピクセル・バッファとデスティネーション・ピクセル・バッファが用意されている．





（a）システム検討，ソフトウェア開発用



（b）LSI検証用

写真1　検証用FPGAボード

（a）のFPGAボードを最初に開発した．この上で，システム・ソフトウェアの開発を行い，ハードウェア化すべき箇所を洗い出した．FPGAには，テレビのNTSCビデオ信号とサウンドを出力するだけのシンプルな論理のみを入れた．（b）のLSI検証用FPGAシステムは，ASICベンダから提供されている評価用チップ（内部のAHBが外部に引き出されている）にユーザ論理を入れるFPGAを接続した．A-Dコンバータ，D-Aコンバータ，一部のIPコアは，同じようにASICベンダから提供されているTEG（test element group）チップを使い，実チップと同等の機能を実現した．評価用チップから出ているAHBは低速でしか動かない．表示用のビデオ帯域を確保するため，ユーザ用外部バスから直接FPGAに接続する専用バイパス・ルートを設けて対策した．

このシステム上にμITRON 4.0（TOPPERS）を移植し，アファイン変換やアルファ・ブレンディング機能を持たせたアニメーション・プレーヤを開発しました．MP3の再生ミドルウェアも移植しました．このシステムでは描画はすべてCPUで行っていたので性能が悪く，どこをハードウェア化すれば対策できるのかが一目瞭然でした．このシステムを解析しながら，GDPをはじめとするLSI仕様の詳細を決めていきました．

本LSIは，Verilog HDLでRTL設計を行い，機能検証は基本的にはレガシなベクタ・シミュレーションによって実施しました．ただし，綿密な検証プランを立て（検証方針を明確化し，検証カバレッジの詳細項目を規定），論理の作り込み（方式設計，コーディング）には細心の注意を払いました．また，**写真1（b）**に示したFPGAボードを作成し，アプリケーション・プログラムの開発と並行にリアルタイム動作としてのシステム検証を実施しました．この結果，ファースト・シリコンで完全動作し，すぐに量産に持っていくことができました．

バックエンド設計（レイアウト設計）においても，できるだけV_{th}（MOSトランジスタのしきい値電圧）が低い高速セルは使わないようにして，消費電力の増加を防ぎました．

● ソフトウェア・インフラの整備も重要

CPUを内蔵したシステムLSIでは，その完成後も，ソフトウェア・インフラの整備が重要になります．本LSIについては，OS（μITRON 4.0，Embedded Linux）やミドルウェア（サウンド，動画，アニメーションなど），各種ドライバ・ソフトウェア（USB，Ethernetなど）を開発しました．

本稿ではARMを内蔵したLSIの開発事例を紹介しました．ARMコアをベースとするシステムLSIの開発のポイントを少しでも感じ取っていただければ幸いです．個人的に毎度のことで恐縮ですが，実際のLSIの開発ではつねに多数の問題に直面し，多くの方に協力をいただきながら，解決して進めてきました．そうした中にたいせつなノウハウが隠されていて，いつも勉強させられています．

参考・引用*文献

(1) ARM；ARM926EJ-S（r0p4/r0p5）Technical Reference Manual, Ref. DD10198D，Jan 2004.
(2) ARM；AMBA Specification Rev2.0，Ref. IHI0011A，May 1999.
(3) ARM；Multi-layer AHB Overview，Ref. DVI0045B，May 2004.
(4) ARM；AHB-Lite Overview，Ref. DVI0044A，Jan 2001.
(5) ARM；ETM9（r2p2）Technical Reference Manual，Ref.DDI 0157F，Aug 2002.
(6) Steve Furber；改訂ARMプロセッサ，CQ出版社，2001年12月.
(7) James D. Foley，Steven K. Feiner，Andries van Dam，John F. Hughes；コンピュータグラフィックス 理論と実践，オーム社，2001年3月.
(8) AP4010の情報，http://www.applause-tech.com/ja_JP/products/ap4010.html

コラム　ARM926コアの動作設定

ARM926コアには，TCMやキャッシュ・メモリ，MMUが内蔵されており，システムの立ち上げ時にはそれらの設定が必要です．この設定は，コア内のシステム制御コプロセッサ（CP15）レジスタを介して行います．CP15レジスタは16本あり，非常に多くの動作設定を行えます（**図A-1**）．

リセット立ち上げ時の手順（スタートアップ・ルーチン処理）の例を以下に示します．

1）処理モードごとのスタック・ポインタの設定
2）ベクタ・エリアやエンディアンなどのCPU基本設定（c1レジスタ）．このとき，命令キャッシュ，データ・キャッシュ，MMUは無効にしておく
3）基本ハードウェア初期化設定（クロックてい倍，外部バス制御の設定など）
4）ITCM/DTCMの設定（c9レジスタ）
5）MMUの変換テーブル（セクション記述子）をメモリ上に作成
6）MMUの設定（c2レジスタ，c3レジスタ）
7）命令キャッシュ，データ・キャッシュ，MMUを有効化（c1レジスタ）
8）データ・キャッシュ，ライト・バッファのクリーンアップ（c7レジスタ）
9）必要があれば（Cコンパイラ関係），RAM上のbssセクション（初期値なし変数エリア）のクリア
10）必要があれば（Cコンパイラ関係），RAM上のdataセクション（初期値付き変数エリア）へのROMからのコピー
11）メイン・プログラム（`main( )`）へのジャンプ

CPU内蔵LSIの機能検証の際には，上記に示したような立ち上げ動作もチェックするので，LSI設計者や検証エンジニアもスタートアップ・ルーチンを作成する必要があります．ARMコアのLSIを設計する場合，CPUそのものの検証は終わっており，個々の命令動作の検証は不要なので，LSIの機能検証用プログラムはC言語を用いて簡単に作成できますが，スタートアップ・ルーチンはアセンブラで記述するので，その習得も必要です．

レジスタ名	リード動作	リード動作
c0	IDコード	禁止
	キャッシュ・タイプ	禁止
	TCMステータス	禁止
c1	CPU制御	
c2	変換テーブル・ベース・アドレス	
c3	ドメイン・アクセス制御	
c4	（リザーブ）	
c5	データ・フォルト・ステータス	
	命令フォルト・ステータス	
c6	フォルト・アドレス	
c7	キャッシュ動作	
c8	禁止	TLB動作
c9	キャッシュ・ロックダウン	
c10	TCM領域	
c11	TLBロックダウン	
c12	（リザーブ）	
c13	（リザーブ）	
	高速コンテキスト・スイッチ拡張プロセスID	
c14	コンテキストID	
c15	テスト用	

（**a**）CP15レジスタ

ARMレジスタからCP15レジスタへの転送命令
```
mcr{cond} p15, <Opcode_1>, <Rd>, <CRn>, <CRm>, <Opcode_2>
```
CP15レジスタからARMレジスタへの転送命令
```
mrc{cond} p15, <Opcode_1>, <Rd>, <CRn>, <CRm>, <Opcode_2>
```

（**b**）アクセス命令

図A-1　システム制御コプロセッサ（CP15）レジスタ

CP15レジスタは`mcr`命令と`mrc`命令でアクセスする．各フィールドの指定方法は，アクセス対象とアクセス方法によって個別に定められている．詳細は英国ARM社の「ARM926EJ-S Technical Reference Manual」を参照のこと．

やまざき・たかなが
（株）アプローズテクノロジーズ
http://www.applause-tech.com/

<筆者プロフィール>
山崎尊永．現在，とあるアルゴリズムを高速実行するためのマルチプロセッサを開発中です．専用プロセッサの開発には，仕込むソフトウェアと合体して目標性能をかならず達成させる責任があり，その開発にはかなり緊張感が漂っています．当面は趣味だと思って楽しもうと思っていますが，そのうちに…？